Application of Information Theory, Lecture 8

Kolmogorov Complexity and Other Entropy Measures

Iftach Haitner

Tel Aviv University.

December 16, 2014

Part I

Kolmogorov Complexity

Description length

- What is the description length of the following strings?

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101
  2. 011010100000100111100110011001111110

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101
  2. 011010100000100111100110011001111110
  3. 111010100110001100111100010101011111

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101
  2. 011010100000100111100110011001111110
  3. 111010100110001100111100010101011111

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101
  2. 011010100000100111100110011001111110
  3. 111010100110001100111100010101011111
- 1. Eighteen 01

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101
  2. 011010100000100111100110011001111110
  3. 111010100110001100111100010101011111

- 1. Eighteen 01
  2. First 36 bit of the binary expansion of $\sqrt{2} - 1$

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101
  2. 011010100000100111100110011001111110
  3. 111010100110001100111100010101011111
- 1. Eighteen 01
  2. First 36 bit of the binary expansion of $\sqrt{2} - 1$
  3. Looks random, but 22 ones out of 36

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101
  2. 011010100000100111100110011001111110
  3. 111010100110001100111100010101011111
- 1. Eighteen 01
  2. First 36 bit of the binary expansion of $\sqrt{2}-1$
  3. Looks random, but 22 ones out of 36
- Bergg's paradox: Let s be "the smallest positive integer that **cannot** be described in twelve English words"

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101
  2. 011010100000100111100110011001111110
  3. 111010100110001100111100010101011111
- 1. Eighteen 01
  2. First 36 bit of the binary expansion of $\sqrt{2} - 1$
  3. Looks random, but 22 ones out of 36
- Bergg's paradox: Let s be "the smallest positive integer that **cannot** be described in twelve English words"
- The above is a definition of s, of less than twelve English words...

Description length

- What is the description length of the following strings?
  1. 010101010101010101010101010101010101
  2. 011010100000100111100110011001111110
  3. 111010100110001100111100010101011111
- 1. Eighteen 01
  2. First 36 bit of the binary expansion of $\sqrt{2} - 1$
  3. Looks random, but 22 ones out of 36
- Bergg's paradox: Let s be "the smallest positive integer that **cannot** be described in twelve English words"
- The above is a definition of s, of less than twelve English words...
- Solution: the word "described" above in the definition of s is not well defined

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term "described" is well defined.

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term "described" is well defined.
- Why C^{++}?

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term "described" is well defined.
- Why C^{++}?
- All (complete) programming language/computational model are essentially equivalent.

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term “described” is well defined.
- Why C^{++}?
- All (complete) programming language/computational model are essentially equivalent.
- Let $K'(x)$ be the description length of x in another complete language, then $|K(x) - k'(x)| \leq const$.

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term “described” is well defined.
- Why C^{++}?
- All (complete) programming language/computational model are essentially equivalent.
- Let $K'(x)$ be the description length of x in another complete language, then $|K(x) - k'(x)| \leq const$.
- What is $K(x)$ for $x = \underbrace{0101010101\ldots01}_{n \text{ pairs}}$

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term "described" is well defined.
- Why C^{++}?
- All (complete) programming language/computational model are essentially equivalent.
- Let $K'(x)$ be the description length of x in another complete language, then $|K(x) - k'(x)| \leq const$.
- What is $K(x)$ for $x = \underbrace{0101010101\ldots01}_{n \text{ pairs}}$
- "For $i = 1 : i^{++} : n$; print 01"

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term "described" is well defined.
- Why C^{++}?
- All (complete) programming language/computational model are essentially equivalent.
- Let $K'(x)$ be the description length of x in another complete language, then $|K(x) - k'(x)| \leq const$.
- What is $K(x)$ for $x = \underbrace{0101010101\ldots01}_{n \text{ pairs}}$
- "For $i = 1 : i^{++} : n$; print 01"
- $K(x) = \log n + const$

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term "described" is well defined.
- Why C^{++}?
- All (complete) programming language/computational model are essentially equivalent.
- Let $K'(x)$ be the description length of x in another complete language, then $|K(x) - k'(x)| \leq const$.
- What is $K(x)$ for $x = \underbrace{0101010101\ldots01}_{n \text{ pairs}}$
- "For $i = 1 : i^{++} : n$; print 01"
- $K(x) = \log n + const$
- This is considered to be small complexity. We typically ignore $\log n$ factors.

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term "described" is well defined.
- Why C^{++}?
- All (complete) programming language/computational model are essentially equivalent.
- Let $K'(x)$ be the description length of x in another complete language, then $|K(x) - k'(x)| \leq const$.
- What is $K(x)$ for $x = \underbrace{0101010101\ldots01}_{n \text{ pairs}}$
- "For $i = 1 : i^{++} : n$; print 01"
- $K(x) = \log n + const$
- This is considered to be small complexity. We typically ignore $\log n$ factors.
- What is $K(x)$ for x being the first n digits of π?

Kolmogorov complexity

- For s string $x \in \{0,1\}^*$, let $K(x)$ be the length of the shortest C^{++} program (written in binary) that outputs x (on empty input)
- Now the term "described" is well defined.
- Why C^{++}?
- All (complete) programming language/computational model are essentially equivalent.
- Let $K'(x)$ be the description length of x in another complete language, then $|K(x) - k'(x)| \leq const$.
- What is $K(x)$ for $x = \underbrace{0101010101\ldots01}_{n\ pairs}$
- "For $i = 1 : i^{++} : n$; print 01"
- $K(x) = \log n + const$
- This is considered to be small complexity. We typically ignore $\log n$ factors.
- What is $K(x)$ for x being the first n digits of π?
- $K(x) = \log n + const$

More examples

More examples

- What is $K(x)$ for $x \in \{0, 1\}^n$ with k ones?

More examples

- What is $K(x)$ for $x \in \{0,1\}^n$ with k ones?
- Recall that $\binom{n}{k} \leq 2^{nH(n/k)}$

More examples

- What is $K(x)$ for $x \in \{0,1\}^n$ with k ones?
- Recall that $\binom{n}{k} \leq 2^{nH(n/k)}$
- Hence $K(x) \leq \log n + nH(n/k)$

Bounds

Bounds

- $K(x) \leq |x| + const$

Bounds

- $K(x) \leq |x| + const$
- Proof: "output "

Bounds

- $K(x) \leq |x| + const$
- Proof: "output "
- Most sequences have high Kolmogorov complexity:

Bounds

- $K(x) \leq |x| + const$
- Proof: "output "
- Most sequences have high Kolmogorov complexity:
- At most 2^{n-1} (C^{++}) programs of length $\leq n - 2$

Bounds

- $K(x) \leq |x| + const$
- Proof: "output "
- Most sequences have high Kolmogorov complexity:
- At most 2^{n-1} (C^{++}) programs of length $\leq n-2$
- 2^n strings of length n

Bounds

- $K(x) \leq |x| + const$
- Proof: "output "
- Most sequences have high Kolmogorov complexity:
- At most 2^{n-1} (C^{++}) programs of length $\leq n-2$
- 2^n strings of length n
- Hence, at least $\frac{1}{2}$ of n-bit strings have Kolmogorov complexity at least $n-1$

Bounds

- $K(x) \leq |x| + const$
- Proof: "output "
- Most sequences have high Kolmogorov complexity:
- At most 2^{n-1} (C^{++}) programs of length $\leq n-2$
- 2^n strings of length n
- Hence, at least $\frac{1}{2}$ of n-bit strings have Kolmogorov complexity at least $n-1$
- In particular, a random sequence has Kolmogorov complexity $\approx n$

Conditional Kolmogorov complexity

Conditional Kolmogorov complexity

- $K(x|y)$ — Kolmogorov complexity of x given y. The length of the shortest partogram that outputd x on input y

Conditional Kolmogorov complexity

- $K(x|y)$ — Kolmogorov complexity of x given y. The length of the shortest partogram that outputd x on input y
- Chain rule (ignoring logs)
  $K(x, y) \approx k(y) + k(X|y)$

H vs. K

H vs. K

$H(X)$ speaks about a random variable X and $K(x)$ of a string x, but

H vs. K

$H(X)$ speaks about a random variable X and $K(x)$ of a string x, but

- Both quantities measure the amount of uncertainty or randomness in an object

H vs. K

$H(X)$ speaks about a random variable X and $K(x)$ of a string x, but

- Both quantities measure the amount of uncertainty or randomness in an object
- Both measure the number of bits it takes to describe an object

H vs. K

$H(X)$ speaks about a random variable X and $K(x)$ of a string x, but

- Both quantities measure the amount of uncertainty or randomness in an object
- Both measure the number of bits it takes to describe an object
- Another property: Let $X_1, \ldots, X_n$ be iid, then whp $K(X) \approx H(X_1, \ldots, X_n) = nH(X_1)$

H vs. K

$H(X)$ speaks about a random variable X and $K(x)$ of a string x, but

- Both quantities measure the amount of uncertainty or randomness in an object
- Both measure the number of bits it takes to describe an object
- Another property: Let $X_1, \ldots, X_n$ be iid, then whp $K(X) \approx H(X_1, \ldots, X_n) = nH(X_1)$
- Proof: ?

H vs. K

$H(X)$ speaks about a random variable X and $K(x)$ of a string x, but

- Both quantities measure the amount of uncertainty or randomness in an object
- Both measure the number of bits it takes to describe an object
- Another property: Let $X_1, \ldots, X_n$ be iid, then whp $K(X) \approx H(X_1, \ldots, X_n) = nH(X_1)$
- Proof: ?

H vs. K

$H(X)$ speaks about a random variable X and $K(x)$ of a string x, but

- Both quantities measure the amount of uncertainty or randomness in an object
- Both measure the number of bits it takes to describe an object
- Another property: Let $X_1, \ldots, X_n$ be iid, then whp $K(X) \approx H(X_1, \ldots, X_n) = nH(X_1)$
- Proof: ? AEP

H vs. K

$H(X)$ speaks about a random variable X and $K(x)$ of a string x, but

- Both quantities measure the amount of uncertainty or randomness in an object
- Both measure the number of bits it takes to describe an object
- Another property: Let $X_1, \ldots, X_n$ be iid, then whp $K(X) \approx H(X_1, \ldots, X_n) = nH(X_1)$
- Proof: ? AEP
- Example: coin flip $(0.7, 0.3)$ then whp we get a string with $K(x) \approx n \cdot h(0.3)$

Universal compression

Universal compression

- A program of length $K(x)$ that outputs x, compresses x into $k(x)$ bit of information.

Universal compression

- A program of length $K(x)$ that outputs x, compresses x into $k(x)$ bit of information.
- Example: length of the human genome: $6 \cdot 10^9$ bits

Universal compression

- A program of length $K(x)$ that outputs x, compresses x into $k(x)$ bit of information.
- Example: length of the human genome: $6 \cdot 10^9$ bits
- But the code is redundant

Universal compression

- A program of length $K(x)$ that outputs x, compresses x into $k(x)$ bit of information.
- Example: length of the human genome: $6 \cdot 10^9$ bits
- But the code is redundant
- The relevant number to measure the number of possible values is the Kolmogorov complexity of the code.

Universal compression

- A program of length $K(x)$ that outputs x, compresses x into $k(x)$ bit of information.
- Example: length of the human genome: $6 \cdot 10^9$ bits
- But the code is redundant
- The relevant number to measure the number of possible values is the Kolmogorov complexity of the code.
- No-one knows its value...

Universal probability

Universal probability

$K(x) = \min_{p\colon p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Universal probability

$K(x) = \min_{p\colon p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p\colon p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty}[p() = x]$

Universal probability

$K(x) = \min_{p\colon p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p\colon p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty} [p() = x]$

- Namely, the probability that if one picks a program at random, it prints x.

Universal probability

$K(x) = \min_{p\colon p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p\colon p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty}[p() = x]$

- Namely, the probability that if one picks a program at random, it prints x.
- Insensitive (up o constant factor) to the computation model.

Universal probability

$K(x) = \min_{p\colon p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p\colon p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty}[p() = x]$

- Namely, the probability that if one picks a program at random, it prints x.
- Insensitive (up o constant factor) to the computation model.
- Interpretation: $P_{\mathcal{U}}(x)$ is the the probability that you observe x in nature.

Universal probability

$K(x) = \min_{p:\ p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p:\ p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty} [p() = x]$

- Namely, the probability that if one picks a program at random, it prints x.
- Insensitive (up o constant factor) to the computation model.
- Interpretation: $P_{\mathcal{U}}(x)$ is the the probability that you observe x in nature.

Universal probability

$K(x) = \min_{p:\ p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p:\ p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty}[p() = x]$

- Namely, the probability that if one picks a program at random, it prints x.
- Insensitive (up o constant factor) to the computation model.
- Interpretation: $P_{\mathcal{U}}(x)$ is the the probability that you observe x in nature.
- Computer as an intelligent amplifier

Universal probability

$K(x) = \min_{p\colon p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p\colon p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty}[p() = x]$

- Namely, the probability that if one picks a program at random, it prints x.
- Insensitive (up o constant factor) to the computation model.
- Interpretation: $P_{\mathcal{U}}(x)$ is the the probability that you observe x in nature.
- Computer as an intelligent amplifier

Theorem 2

$\exists c > 0$ *such that* $2^{-K(x)} \leq P_{\mathcal{U}}(x) \leq c \cdot 2^{-K(x)}$ *for every* $x \in \{0,1\}^*$.

Universal probability

$K(x) = \min_{p\colon p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p\colon p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty}[p() = x]$

- Namely, the probability that if one picks a program at random, it prints x.
- Insensitive (up o constant factor) to the computation model.
- Interpretation: $P_{\mathcal{U}}(x)$ is the the probability that you observe x in nature.
- Computer as an intelligent amplifier

Theorem 2

$\exists c > 0$ such that $2^{-K(x)} \leq P_{\mathcal{U}}(x) \leq c \cdot 2^{-K(x)}$ for every $x \in \{0,1\}^*$.

Universal probability

$K(x) = \min_{p\colon p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p\colon p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty}[p() = x]$

- Namely, the probability that if one picks a program at random, it prints x.
- Insensitive (up o constant factor) to the computation model.
- Interpretation: $P_{\mathcal{U}}(x)$ is the the probability that you observe x in nature.
- Computer as an intelligent amplifier

Theorem 2

$\exists c > 0$ such that $2^{-K(x)} \le P_{\mathcal{U}}(x) \le c \cdot 2^{-K(x)}$ for every $x \in \{0,1\}^*$.

- The interesting part is $P_{\mathcal{U}}(x) \le c \cdot 2^{-K(x)}$

Universal probability

$K(x) = \min_{p:\ p()=x} |p|$, where $p()$ is the output of C^{++} program defined by p.

Definition 1

The universal probability of a string x is
$P_{\mathcal{U}}(x) = \sum_{p:\ p()=x} 2^{-|p|} = \Pr_{p \leftarrow \{0,1\}^\infty}[p() = x]$

- Namely, the probability that if one picks a program at random, it prints x.
- Insensitive (up o constant factor) to the computation model.
- Interpretation: $P_{\mathcal{U}}(x)$ is the the probability that you observe x in nature.
- Computer as an intelligent amplifier

Theorem 2

$\exists c > 0$ such that $2^{-K(x)} \leq P_{\mathcal{U}}(x) \leq c \cdot 2^{-K(x)}$ for every $x \in \{0,1\}^*$.

- The interesting part is $P_{\mathcal{U}}(x) \leq c \cdot 2^{-K(x)}$
- Hence, for $X \sim P_{\mathcal{U}}$, it holds that $|K(X) - H(X)| \leq c$!

Proving Theorem 2

Proving Theorem 2

- We need to find $c > 0$ such that $k(x) \leq \log \frac{1}{P_{\mathcal{U}}(x)} + c$ for every $x \in \{0,1\}^*$

Proving Theorem 2

- We need to find $c > 0$ such that $k(x) \leq \log \frac{1}{P_{\mathcal{U}}(x)} + c$ for every $x \in \{0,1\}^*$
- In other words, find a program to output x whose length is $\log \frac{1}{P_{\mathcal{U}}(x)} + c$

Proving Theorem 2

- We need to find $c > 0$ such that $k(x) \leq \log \frac{1}{P_{\mathcal{U}}(x)} + c$ for every $x \in \{0,1\}^*$
- In other words, find a program to output x whose length is $\log \frac{1}{P_{\mathcal{U}}(x)} + c$
- Idea, program chooses a leaf on the Shannon code for $P_{\mathcal{U}}$ (in which x is of depth $\left\lceil \log \frac{1}{P_{\mathcal{U}}(x)} \right\rceil$)

Proving Theorem 2

- We need to find $c > 0$ such that $k(x) \leq \log \frac{1}{P_{\mathcal{U}}(x)} + c$ for every $x \in \{0,1\}^*$
- In other words, find a program to output x whose length is $\log \frac{1}{P_{\mathcal{U}}(x)} + c$
- Idea, program chooses a leaf on the Shannon code for $P_{\mathcal{U}}$ (in which x is of depth $\left\lceil \log \frac{1}{P_{\mathcal{U}}(x)} \right\rceil$)
- Problem: $P_{\mathcal{U}}$ is not computable

Proving Theorem 2

- We need to find $c > 0$ such that $k(x) \leq \log \frac{1}{P_{\mathcal{U}}(x)} + c$ for every $x \in \{0,1\}^*$
- In other words, find a program to output x whose length is $\log \frac{1}{P_{\mathcal{U}}(x)} + c$
- Idea, program chooses a leaf on the Shannon code for $P_{\mathcal{U}}$ (in which x is of depth $\left\lceil \log \frac{1}{P_{\mathcal{U}}(x)} \right\rceil$)
- Problem: $P_{\mathcal{U}}$ is not computable
- Solution: compute a better and better estimate for the tree of $P_{\mathcal{U}}$ along with the “mapping" from the tree nodes back to codewords.

Proving Theorem 2

Proving Theorem 2

- Initial T to be the infinite Binary tree.

Proving Theorem 2

- Initial T to be the infinite Binary tree.

Program 3 (M)

Enumerate over all programs in $\{0,1\}^*$: at round i run the first i programs (one after the other), for i steps, and do: If program p outputs a string x and $(*, x, n(x)) \notin T$, place $(p, x, n(x))$ at unused $n(x)$-depth node of T, for $n(x) = \left\lceil \log \frac{1}{\hat{P}_{\mathcal{U}}(x)} \right\rceil + 1$ and $\hat{P}_{\mathcal{U}}(x) = \sum_{(p', x, \cdot) \in T:\ p'() = x} 2^{-|p'|}$

Proving Theorem 2

- Initial T to be the infinite Binary tree.

Program 3 (M)

Enumerate over all programs in $\{0,1\}^*$: at round i run the first i programs (one after the other), for i steps, and do: If program p outputs a string x and $(*, x, n(x)) \notin T$, place $(p, x, n(x))$ at unused $n(x)$-depth node of T, for $n(x) = \left\lceil \log \frac{1}{\hat{P}_{\mathcal{U}}(x)} \right\rceil + 1$ and $\hat{P}_{\mathcal{U}}(x) = \sum_{(p',x,\cdot)\in T:\ p'()=x} 2^{-|p'|}$

- The program never gets stack (can always add the node).

Proving Theorem 2

- Initial T to be the infinite Binary tree.

Program 3 (M)

Enumerate over all programs in $\{0,1\}^*$: at round i run the first i programs (one after the other), for i steps, and do: If program p outputs a string x and $(*, x, n(x)) \notin T$, place $(p, x, n(x))$ at unused $n(x)$-depth node of T, for $n(x) = \left\lceil \log \frac{1}{\hat{P}_{\mathcal{U}}(x)} \right\rceil + 1$ and $\hat{P}_{\mathcal{U}}(x) = \sum_{(p',x,\cdot)\in T:\ p'()=x} 2^{-|p'|}$

- The program never gets stack (can always add the node).

  Proof: Let $x \in \{0,1\}^*$. At each point through the execution of M, $\sum_{(p,x,\cdot)\in T} 2^{-|p|} \leq 2^{-K(x)}$

Proving Theorem 2

- Initial T to be the infinite Binary tree.

Program 3 (M)

Enumerate over all programs in $\{0,1\}^*$: at round i run the first i programs (one after the other), for i steps, and do: If program p outputs a string x and $(*, x, n(x)) \notin T$, place $(p, x, n(x))$ at unused $n(x)$-depth node of T, for $n(x) = \left\lceil \log \frac{1}{\hat{P}_{\mathcal{U}}(x)} \right\rceil + 1$ and $\hat{P}_{\mathcal{U}}(x) = \sum_{(p', x, \cdot) \in T:\ p'() = x} 2^{-|p'|}$

- The program never gets stack (can always add the node).

  Proof: Let $x \in \{0,1\}^*$. At each point through the execution of M, $\sum_{(p,x,\cdot) \in T} 2^{-|p|} \leq 2^{-K(x)}$

  Since $\sum_x 2^{-K(x)} \leq 1$, the proof follows by Kraft inequality.

Proving Theorem 2

- Initial T to be the infinite Binary tree.

Program 3 (M)

Enumerate over all programs in $\{0,1\}^*$: at round i run the first i programs (one after the other), for i steps, and do: If program p outputs a string x and $(*, x, n(x)) \notin T$, place $(p, x, n(x))$ at unused $n(x)$-depth node of T, for $n(x) = \left\lceil \log \frac{1}{\hat{P}_{\mathcal{U}}(x)} \right\rceil + 1$ and $\hat{P}_{\mathcal{U}}(x) = \sum_{(p',x,\cdot)\in T:\ p'()=x} 2^{-|p'|}$

- The program never gets stack (can always add the node).

  Proof: Let $x \in \{0,1\}^*$. At each point through the execution of M, $\sum_{(p,x,\cdot)\in T} 2^{-|p|} \leq 2^{-K(x)}$

  Since $\sum_x 2^{-K(x)} \leq 1$, the proof follows by Kraft inequality.

- $\forall x \in \{0,1\}^*$: M adds a node $(\cdot, x, \cdot)$ to T at depth $1 + \left\lceil \log \frac{1}{P_{\mathcal{U}}(x)} \right\rceil$

Proving Theorem 2

- Initial T to be the infinite Binary tree.

Program 3 (M)

Enumerate over all programs in $\{0,1\}^*$: at round i run the first i programs (one after the other), for i steps, and do: If program p outputs a string x and $(*, x, n(x)) \notin T$, place $(p, x, n(x))$ at unused $n(x)$-depth node of T, for $n(x) = \left\lceil \log \frac{1}{\hat{P}_{\mathcal{U}}(x)} \right\rceil + 1$ and $\hat{P}_{\mathcal{U}}(x) = \sum_{(p',x,\cdot)\in T:\ p'()=x} 2^{-|p'|}$

- The program never gets stack (can always add the node).

  Proof: Let $x \in \{0,1\}^*$. At each point through the execution of M, $\sum_{(p,x,\cdot)\in T} 2^{-|p|} \leq 2^{-K(x)}$

  Since $\sum_x 2^{-K(x)} \leq 1$, the proof follows by Kraft inequality.

- $\forall x \in \{0,1\}^*$: M adds a node $(\cdot, x, \cdot)$ to T at depth $1 + \left\lceil \log \frac{1}{P_{\mathcal{U}}(x)} \right\rceil$

  Proof: $\hat{P}_{\mathcal{U}}(x)$ converges to $P_{\mathcal{U}}(x)$

Proving Theorem 2

- Initial T to be the infinite Binary tree.

Program 3 (M)

Enumerate over all programs in $\{0,1\}^*$: at round i run the first i programs (one after the other), for i steps, and do: If program p outputs a string x and $(*, x, n(x)) \notin T$, place $(p, x, n(x))$ at unused $n(x)$-depth node of T, for $n(x) = \left\lceil \log \frac{1}{\hat{P}_{\mathcal{U}}(x)} \right\rceil + 1$ and $\hat{P}_{\mathcal{U}}(x) = \sum_{(p',x,\cdot)\in T:\ p'()=x} 2^{-|p'|}$

- The program never gets stack (can always add the node).

  Proof: Let $x \in \{0,1\}^*$. At each point through the execution of M, $\sum_{(p,x,\cdot)\in T} 2^{-|p|} \leq 2^{-K(x)}$

  Since $\sum_x 2^{-K(x)} \leq 1$, the proof follows by Kraft inequality.

- $\forall x \in \{0,1\}^*$: M adds a node $(\cdot, x, \cdot)$ to T at depth $1 + \left\lceil \log \frac{1}{P_{\mathcal{U}}(x)} \right\rceil$

  Proof: $\hat{P}_{\mathcal{U}}(x)$ converges to $P_{\mathcal{U}}(x)$

- For $x \in \{0,1\}^*$, let $\ell(x)$ be the location its $(1 + \left\lceil \log \frac{1}{P_{\mathcal{U}}(x)} \right\rceil)$-depth node

Proving Theorem 2

- Initial T to be the infinite Binary tree.

Program 3 (M)

Enumerate over all programs in $\{0,1\}^*$: at round i run the first i programs (one after the other), for i steps, and do: If program p outputs a string x and $(*, x, n(x)) \notin T$, place $(p, x, n(x))$ at unused $n(x)$-depth node of T, for $n(x) = \left\lceil \log \frac{1}{\hat{P}_{\mathcal{U}}(x)} \right\rceil + 1$ and $\hat{P}_{\mathcal{U}}(x) = \sum_{(p',x,\cdot)\in T:\ p'()=x} 2^{-|p'|}$

- The program never gets stack (can always add the node).

  Proof: Let $x \in \{0,1\}^*$. At each point through the execution of M, $\sum_{(p,x,\cdot)\in T} 2^{-|p|} \leq 2^{-K(x)}$

  Since $\sum_x 2^{-K(x)} \leq 1$, the proof follows by Kraft inequality.

- $\forall x \in \{0,1\}^*$: M adds a node $(\cdot, x, \cdot)$ to T at depth $1 + \left\lceil \log \frac{1}{P_{\mathcal{U}}(x)} \right\rceil$

  Proof: $\hat{P}_{\mathcal{U}}(x)$ converges to $P_{\mathcal{U}}(x)$

- For $x \in \{0,1\}^*$, let $\ell(x)$ be the location its $(1 + \left\lceil \log \frac{1}{P_{\mathcal{U}}(x)} \right\rceil)$-depth node
- Program for printing x. Run M till it assigns the node at the location of $\ell(x)$

Applications

Applications

- (another) Proof that there are infinity many primes.

Applications

- (another) Proof that there are infinity many primes.
- Assume there are finitely many primes $p_1, \ldots, p_m$

Applications

- (another) Proof that there are infinity many primes.
- Assume there are finitely many primes $p_1, \ldots, p_m$
- Any length n integer x can be written as $x = \prod_{i=1}^{m} p_i^{d_i}$

Applications

- (another) Proof that there are infinity many primes.
- Assume there are finitely many primes $p_1, \ldots, p_m$
- Any length n integer x can be written as $x = \prod_{i=1}^{m} p_i^{d_i}$
- $d_i \leq n$, hence length $d_i \leq \log n$

Applications

- (another) Proof that there are infinity many primes.
- Assume there are finitely many primes $p_1, \ldots, p_m$
- Any length n integer x can be written as $x = \prod_{i=1}^{m} p_i^{d_i}$
- $d_i \leq n$, hence length $d_i \leq \log n$
- Hence, $K(x) \leq m \cdot \log n + const$

Applications

- (another) Proof that there are infinity many primes.
- Assume there are finitely many primes $p_1, \ldots, p_m$
- Any length n integer x can be written as $x = \prod_{i=1}^{m} p_i^{d_i}$
- $d_i \leq n$, hence length $d_i \leq \log n$
- Hence, $K(x) \leq m \cdot \log n + const$
- But for most numbers $k(x) \geq n - 1$

Computability of K

Computability of K

- Can we compute $K(x)$?

Computability of K

- Can we compute $K(x)$?
- Answer, No.

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C
- Let s be the smallest positive integer s.t. $K(s) > 2C + 10,000$

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C
- Let s be the smallest positive integer s.t. $K(s) > 2C + 10,000$
- s can be computed by the following program:

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C
- Let s be the smallest positive integer s.t. $K(s) > 2C + 10,000$
- s can be computed by the following program:
  1. $x = 0$

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C
- Let s be the smallest positive integer s.t. $K(s) > 2C + 10,000$
- s can be computed by the following program:
  1. $x = 0$
  2. While $(K(x) < 2C + 10,000)$: x^{++}

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C
- Let s be the smallest positive integer s.t. $K(s) > 2C + 10,000$
- s can be computed by the following program:
  1. $x = 0$
  2. While $(K(x) < 2C + 10,000)$: x^{++}
  3. Output x

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C
- Let s be the smallest positive integer s.t. $K(s) > 2C + 10,000$
- s can be computed by the following program:
  1. $x = 0$
  2. While $(K(x) < 2C + 10,000)$: x^{++}
  3. Output x
- Thus $K(s) < C + \log C + \log 10,000 + const < 2C + 10,000$

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C
- Let s be the smallest positive integer s.t. $K(s) > 2C + 10,000$
- s can be computed by the following program:
  1. $x = 0$
  2. While $(K(x) < 2C + 10,000)$: x^{++}
  3. Output x
- Thus $K(s) < C + \log C + \log 10,000 + const < 2C + 10,000$
- Bergg's Paradox, revisited:

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C
- Let s be the smallest positive integer s.t. $K(s) > 2C + 10,000$
- s can be computed by the following program:
  1. $x = 0$
  2. While $(K(x) < 2C + 10,000)$: x^{++}
  3. Output x
- Thus $K(s) < C + \log C + \log 10,000 + const < 2C + 10,000$
- Bergg's Paradox, revisited:
- s — the smallest positive number with $K(s) > 10000$

Computability of K

- Can we compute $K(x)$?
- Answer, No.
- Proof: Assume K is computable by a program of length C
- Let s be the smallest positive integer s.t. $K(s) > 2C + 10,000$
- s can be computed by the following program:
  1. $x = 0$
  2. While $(K(x) < 2C + 10,000)$: x^{++}
  3. Output x
- Thus $K(s) < C + \log C + \log 10,000 + const < 2C + 10,000$
- Bergg's Paradox, revisited:
- s — the smallest positive number with $K(s) > 10000$
- This is not a paradox, since the description of s is not short.

Explicit large complexity strings

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

- For most strings $K(x) > C + 1$, but it cannot be proven even for a single string

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

- For most strings $K(x) > C + 1$, but it cannot be proven even for a single string
- $K(x) \geq C$ is an example for a theorem that cannot be proven, and for most x's cannot be disproved.

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

- For most strings $K(x) > C + 1$, but it cannot be proven even for a single string
- $K(x) \geq C$ is an example for a theorem that cannot be proven, and for most x's cannot be disproved.
- Proof: for integer C define the program T_C:

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

- For most strings $K(x) > C + 1$, but it cannot be proven even for a single string
- $K(x) \geq C$ is an example for a theorem that cannot be proven, and for most x's cannot be disproved.
- Proof: for integer C define the program T_C:
  1. $y = 0$

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

- For most strings $K(x) > C + 1$, but it cannot be proven even for a single string
- $K(x) \geq C$ is an example for a theorem that cannot be proven, and for most x's cannot be disproved.
- Proof: for integer C define the program T_C:
  1. $y = 0$
  2. If y is a proof for the statement $k(x) > C$, output x

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

- For most strings $K(x) > C + 1$, but it cannot be proven even for a single string
- $K(x) \geq C$ is an example for a theorem that cannot be proven, and for most x's cannot be disproved.
- Proof: for integer C define the program T_C:
  1. $y = 0$
  2. If y is a proof for the statement $k(x) > C$, output x
  3. y^{++}

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

- For most strings $K(x) > C + 1$, but it cannot be proven even for a single string
- $K(x) \geq C$ is an example for a theorem that cannot be proven, and for most x's cannot be disproved.
- Proof: for integer C define the program T_C:
  1. $y = 0$
  2. If y is a proof for the statement $k(x) > C$, output x
  3. y^{++}
- $|T_C| = \log C + D$, where D is a const

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

- For most strings $K(x) > C + 1$, but it cannot be proven even for a single string
- $K(x) \geq C$ is an example for a theorem that cannot be proven, and for most x's cannot be disproved.
- Proof: for integer C define the program T_C:
  1. $y = 0$
  2. If y is a proof for the statement $k(x) > C$, output x
  3. y^{++}
- $|T_C| = \log C + D$, where D is a const
- Take C such that $C > \log C + D$

Explicit large complexity strings

- Can we give an explicit example of string x with large $k(x)$?

Theorem 4

$\exists$ constant C s.t. $\forall$ longer than C string x, the theorem $K(x) \geq C$ cannot be proven (under any reasonable axiom system).

- For most strings $K(x) > C + 1$, but it cannot be proven even for a single string
- $K(x) \geq C$ is an example for a theorem that cannot be proven, and for most x's cannot be disproved.
- Proof: for integer C define the program T_C:
  1. $y = 0$
  2. If y is a proof for the statement $k(x) > C$, output x
  3. y^{++}
- $|T_C| = \log C + D$, where D is a const
- Take C such that $C > \log C + D$
- If T_C stops and outputs x, then $k(x) < \log C + D < C$, a contradiction to the fact that $\exists$ proof that $k(x) > C$.

Part II

Other Entropy Measures

Other entropy measures

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X[-\log p(X)]$

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X[-\log p(X)]$
- Max entropy of X is $\mathrm{H}_0(X) = \log |\mathrm{Supp}(X)|$

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log |\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log |\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log |\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X[-\log p(X)]$
- Max entropy of X is $\mathrm{H}_0(X) = \log |\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$

  Probability of collision when drawing two independent samples from X

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log|\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$
  Probability of collision when drawing two independent samples from X
- Collision entropy/Renyi entropy of X is $\mathrm{H}_2(X) = -\log \mathrm{CP}(X)$

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x\in\mathcal{X}} -p(x)\cdot\log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log|\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x\in\mathcal{X}}\{-\log p(x)\} = -\log\max_{x\in\mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x\in\mathcal{X}} p(x)^2$
  Probability of collision when drawing two independent samples from X
- Collision entropy/Renyi entropy of X is $\mathrm{H}_2(X) = -\log\mathrm{CP}(X)$
- $\mathrm{H}_\infty(X) \le \mathrm{H}_2(X) \le H(X) \le \mathrm{H}_0(X)$ (Jensen)
  Equality iff X is uniform over $\mathcal{X}$

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log |\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$
  Probability of collision when drawing two independent samples from X
- Collision entropy/Renyi entropy of X is $\mathrm{H}_2(X) = -\log \mathrm{CP}(X)$
- $\mathrm{H}_\infty(X) \leq \mathrm{H}_2(X) \leq H(X) \leq \mathrm{H}_0(X)$ (Jensen)
  Equality iff X is uniform over $\mathcal{X}$
- For instance, $\mathrm{CP}(X) \leq \sum_x p(x) \max_{x'} p(x') = \max_{x'} p(x')$.

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log|\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$
  Probability of collision when drawing two independent samples from X
- Collision entropy/Renyi entropy of X is $\mathrm{H}_2(X) = -\log \mathrm{CP}(X)$
- $\mathrm{H}_\infty(X) \leq \mathrm{H}_2(X) \leq H(X) \leq \mathrm{H}_0(X)$ (Jensen)
  Equality iff X is uniform over $\mathcal{X}$
- For instance, $\mathrm{CP}(X) \leq \sum_x p(x) \max_{x'} p(x') = \max_{x'} p(x')$.

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log|\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$
  Probability of collision when drawing two independent samples from X
- Collision entropy/Renyi entropy of X is $\mathrm{H}_2(X) = -\log \mathrm{CP}(X)$
- $\mathrm{H}_\infty(X) \leq \mathrm{H}_2(X) \leq H(X) \leq \mathrm{H}_0(X)$ (Jensen)
  Equality iff X is uniform over $\mathcal{X}$
- For instance, $\mathrm{CP}(X) \leq \sum_x p(x) \max_{x'} p(x') = \max_{x'} p(x')$. Hence, $\mathrm{H}_2(X) \geq -\log \max_{x'} p(x') = \mathrm{H}_\infty(X)$.

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x\in\mathcal{X}} -p(x)\cdot\log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log|\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x\in\mathcal{X}}\{-\log p(x)\} = -\log\max_{x\in\mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x\in\mathcal{X}} p(x)^2$
  Probability of collision when drawing two independent samples from X
- Collision entropy/Renyi entropy of X is $\mathrm{H}_2(X) = -\log\mathrm{CP}(X)$
- $\mathrm{H}_\infty(X) \le \mathrm{H}_2(X) \le H(X) \le \mathrm{H}_0(X)$ (Jensen)
  Equality iff X is uniform over $\mathcal{X}$
- For instance, $\mathrm{CP}(X) \le \sum_x p(x)\max_{x'} p(x') = \max_{x'} p(x')$. Hence, $\mathrm{H}_2(X) \ge -\log\max_{x'} p(x') = \mathrm{H}_\infty(X)$.
- $\mathrm{H}_2(X) \le 2\,\mathrm{H}_\infty(X)$

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log |\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$
  Probability of collision when drawing two independent samples from X
- Collision entropy/Renyi entropy of X is $\mathrm{H}_2(X) = -\log \mathrm{CP}(X)$
- $\mathrm{H}_\infty(X) \le \mathrm{H}_2(X) \le H(X) \le \mathrm{H}_0(X)$ (Jensen)
  Equality iff X is uniform over $\mathcal{X}$
- For instance, $\mathrm{CP}(X) \le \sum_x p(x) \max_{x'} p(x') = \max_{x'} p(x')$. Hence, $\mathrm{H}_2(X) \ge -\log \max_{x'} p(x') = \mathrm{H}_\infty(X)$.
- $\mathrm{H}_2(X) \le 2\,\mathrm{H}_\infty(X)$
- Proof: $\mathrm{CP}(X) \ge (\max_{x'} p(x'))^2$.

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log |\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$
  Probability of collision when drawing two independent samples from X
- Collision entropy/Renyi entropy of X is $\mathrm{H}_2(X) = -\log \mathrm{CP}(X)$
- $\mathrm{H}_\infty(X) \leq \mathrm{H}_2(X) \leq H(X) \leq \mathrm{H}_0(X)$ (Jensen)
  Equality iff X is uniform over $\mathcal{X}$
- For instance, $\mathrm{CP}(X) \leq \sum_x p(x) \max_{x'} p(x') = \max_{x'} p(x')$. Hence, $\mathrm{H}_2(X) \geq -\log \max_{x'} p(x') = \mathrm{H}_\infty(X)$.
- $\mathrm{H}_2(X) \leq 2\,\mathrm{H}_\infty(X)$
- Proof: $\mathrm{CP}(X) \geq (\max_{x'} p(x'))^2$.

Other entropy measures

Let $X \sim p$ be a random variable over $\mathcal{X}$.

- Recall that Shannon entropy of X is $H(X) = H_1(X) = \sum_{x \in \mathcal{X}} -p(x) \cdot \log p(x) = \mathrm{E}_X\left[-\log p(X)\right]$
- Max entropy of X is $\mathrm{H}_0(X) = \log |\mathrm{Supp}(X)|$
- Min entropy of X is $\mathrm{H}_\infty(X) = \min_{x \in \mathcal{X}}\{-\log p(x)\} = -\log \max_{x \in \mathcal{X}}\{p(x)\}$
- Collision probability of X is $\mathrm{CP}(X) = \sum_{x \in \mathcal{X}} p(x)^2$
  Probability of collision when drawing two independent samples from X
- Collision entropy/Renyi entropy of X is $\mathrm{H}_2(X) = -\log \mathrm{CP}(X)$
- $\mathrm{H}_\infty(X) \leq \mathrm{H}_2(X) \leq H(X) \leq \mathrm{H}_0(X)$ (Jensen)
  Equality iff X is uniform over $\mathcal{X}$
- For instance, $\mathrm{CP}(X) \leq \sum_x p(x) \max_{x'} p(x') = \max_{x'} p(x')$. Hence, $\mathrm{H}_2(X) \geq -\log \max_{x'} p(x') = \mathrm{H}_\infty(X)$.
- $\mathrm{H}_2(X) \leq 2\,\mathrm{H}_\infty(X)$
- Proof: $\mathrm{CP}(X) \geq (\max_{x'} p(x'))^2$. Hence, $-\log \mathrm{CP}(X) \leq -2\,\mathrm{H}_\infty(X)$

Other entropy measures, cont

Other entropy measures, cont

- No simple chain rule.

Other entropy measures, cont

- No simple chain rule.
- Let $X = \perp$ wp $\frac{1}{2}$ and uniform over $\{0,1\}^n$ otherwise, and let Y be indicator for $X = \perp$.

Other entropy measures, cont

- No simple chain rule.
- Let $X = \perp$ wp $\frac{1}{2}$ and uniform over $\{0,1\}^n$ otherwise, and let Y be indicator for $X = \perp$.
- $\mathrm{H}_\infty(X|Y=1) = 0$ and $\mathrm{H}_\infty(X|Y=0) = n$. But $\mathrm{H}_\infty(X) = 1$.

Section 1

Shannon to Min entropy

Shannon to Min entropy

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ *and let* $\varepsilon > 0$*. Then* $\Pr\left[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)\right] < 2 \cdot e^{-2\varepsilon^2 n}$.

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ *and let* $\varepsilon > 0$. *Then* $\Pr\left[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)\right] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof:

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ *and let* $\varepsilon > 0$. *Then* $\Pr\left[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)\right] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ *and let* $\varepsilon > 0$. *Then* $\Pr\left[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)\right] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

- $A_{n,\varepsilon} := \{\mathbf{x} \in \mathrm{Supp}(X^n) \colon 2^{-n(H(X)+\varepsilon)} \leq p^n(\mathbf{x}) \leq 2^{-n(H(X)-\varepsilon)}\}$

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ and let $\varepsilon > 0$. Then $\Pr\left[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)\right] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

- $A_{n,\varepsilon} := \{\mathbf{x} \in \mathrm{Supp}(X^n) \colon 2^{-n(H(X)+\varepsilon)} \leq p^n(\mathbf{x}) \leq 2^{-n(H(X)-\varepsilon)}\}$
- $-\log p^n(\mathbf{x}) \geq n \cdot (\mathrm{H}(X) - \varepsilon)$ for any $\mathbf{x} \in A_{n,\varepsilon}$

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ and let $\varepsilon > 0$. Then $\Pr\left[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)\right] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

- $A_{n,\varepsilon} := \{\mathbf{x} \in \mathrm{Supp}(X^n) \colon 2^{-n(H(X)+\varepsilon)} \leq p^n(\mathbf{x}) \leq 2^{-n(H(X)-\varepsilon)}\}$
- $-\log p^n(\mathbf{x}) \geq n \cdot (\mathrm{H}(X) - \varepsilon)$ for any $\mathbf{x} \in A_{n,\varepsilon}$

Proposition 6 (Hoeffding's inequality)

Let $Z^1, \ldots, Z^n$ be iids over $[0, 1]$ with expectation μ. Then,
$\Pr\left[\left|\frac{\sum_{j=i}^n Z^j}{n} - \mu\right| \geq \varepsilon\right] \leq 2 \cdot e^{-2\varepsilon^2 n}$ for every $\varepsilon > 0$.

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ and let $\varepsilon > 0$. Then $\Pr[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

- $A_{n,\varepsilon} := \{\mathbf{x} \in \mathrm{Supp}(X^n) \colon 2^{-n(H(X)+\varepsilon)} \leq p^n(\mathbf{x}) \leq 2^{-n(H(X)-\varepsilon)}\}$
- $-\log p^n(\mathbf{x}) \geq n \cdot (\mathrm{H}(X) - \varepsilon)$ for any $\mathbf{x} \in A_{n,\varepsilon}$

Proposition 6 (Hoeffding's inequality)

Let $Z^1, \ldots, Z^n$ be iids over $[0,1]$ with expectation μ. Then,
$\Pr\left[\left|\frac{\sum_{j=i}^n Z^j}{n} - \mu\right| \geq \varepsilon\right] \leq 2 \cdot e^{-2\varepsilon^2 n}$ for every $\varepsilon > 0$.

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ and let $\varepsilon > 0$. Then $\Pr[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

- $A_{n,\varepsilon} := \{\mathbf{x} \in \mathrm{Supp}(X^n) \colon 2^{-n(H(X)+\varepsilon)} \leq p^n(\mathbf{x}) \leq 2^{-n(H(X)-\varepsilon)}\}$
- $-\log p^n(\mathbf{x}) \geq n \cdot (\mathrm{H}(X) - \varepsilon)$ for any $\mathbf{x} \in A_{n,\varepsilon}$

Proposition 6 (Hoeffding's inequality)

Let $Z^1, \ldots, Z^n$ be iids over $[0, 1]$ with expectation μ. Then, $\Pr\left[\left|\frac{\sum_{j=i}^n Z^j}{n} - \mu\right| \geq \varepsilon\right] \leq 2 \cdot e^{-2\varepsilon^2 n}$ for every $\varepsilon > 0$.

- Taking $Z_i = \log p(X_i)$, it follows that $\Pr[X^n \notin A_{n,\varepsilon}] \leq 2 \cdot e^{-2\varepsilon^2 n}$

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ and let $\varepsilon > 0$. Then $\Pr[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

- $A_{n,\varepsilon} := \{\mathbf{x} \in \mathrm{Supp}(X^n) \colon 2^{-n(H(X)+\varepsilon)} \leq p^n(\mathbf{x}) \leq 2^{-n(H(X)-\varepsilon)}\}$
- $-\log p^n(\mathbf{x}) \geq n \cdot (\mathrm{H}(X) - \varepsilon)$ for any $\mathbf{x} \in A_{n,\varepsilon}$

Proposition 6 (Hoeffding's inequality)

Let $Z^1, \ldots, Z^n$ be iids over $[0,1]$ with expectation μ. Then,
$\Pr\left[\left|\frac{\sum_{j=i}^n Z^j}{n} - \mu\right| \geq \varepsilon\right] \leq 2 \cdot e^{-2\varepsilon^2 n}$ for every $\varepsilon > 0$.

- Taking $Z_i = \log p(X_i)$, it follows that $\Pr[X^n \notin A_{n,\varepsilon}] \leq 2 \cdot e^{-2\varepsilon^2 n}$

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ and let $\varepsilon > 0$. Then $\Pr\left[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)\right] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

- $A_{n,\varepsilon} := \{\mathbf{x} \in \mathrm{Supp}(X^n) \colon 2^{-n(H(X)+\varepsilon)} \leq p^n(\mathbf{x}) \leq 2^{-n(H(X)-\varepsilon)}\}$
- $-\log p^n(\mathbf{x}) \geq n \cdot (\mathrm{H}(X) - \varepsilon)$ for any $\mathbf{x} \in A_{n,\varepsilon}$

Proposition 6 (Hoeffding's inequality)

Let $Z^1, \ldots, Z^n$ be iids over $[0,1]$ with expectation μ. Then, $\Pr\left[\left|\frac{\sum_{j=i}^n Z^j}{n} - \mu\right| \geq \varepsilon\right] \leq 2 \cdot e^{-2\varepsilon^2 n}$ for every $\varepsilon > 0$.

- Taking $Z_i = \log p(X_i)$, it follows that $\Pr\left[X^n \notin A_{n,\varepsilon}\right] \leq 2 \cdot e^{-2\varepsilon^2 n}$

Corollary 7

$\exists$ rv W that is $(2 \cdot e^{-2\varepsilon^2 n})$-close to X^n, and $\mathrm{H}_\infty(W) \geq n(H(X) - \varepsilon)$.

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ and let $\varepsilon > 0$. Then $\Pr\left[-\log p^n(X^n) \leq n \cdot (\mathrm{H}(X) - \varepsilon)\right] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

- $A_{n,\varepsilon} := \{\mathbf{x} \in \mathrm{Supp}(X^n) \colon 2^{-n(H(X)+\varepsilon)} \leq p^n(\mathbf{x}) \leq 2^{-n(H(X)-\varepsilon)}\}$
- $-\log p^n(\mathbf{x}) \geq n \cdot (\mathrm{H}(X) - \varepsilon)$ for any $\mathbf{x} \in A_{n,\varepsilon}$

Proposition 6 (Hoeffding's inequality)

Let $Z^1, \ldots, Z^n$ be iids over $[0, 1]$ with expectation μ. Then, $\Pr\left[\left|\frac{\sum_{j=i}^n Z^j}{n} - \mu\right| \geq \varepsilon\right] \leq 2 \cdot e^{-2\varepsilon^2 n}$ for every $\varepsilon > 0$.

- Taking $Z_i = \log p(X_i)$, it follows that $\Pr\left[X^n \notin A_{n,\varepsilon}\right] \leq 2 \cdot e^{-2\varepsilon^2 n}$

Corollary 7

$\exists$ rv W that is $(2 \cdot e^{-2\varepsilon^2 n})$-close to X^n, and $\mathrm{H}_\infty(W) \geq n(H(X) - \varepsilon)$.

Proof:

Shannon to Min entropy

Given rv $X \sim p$, let X^n denote n independent copies of X, and let $p^n(x_1 \ldots, x_n) = \prod_{i=1}^n p(x_i)$.

Lemma 5

Let $X \sim p$ and let $\varepsilon > 0$. Then $\Pr\left[-\log p^n(X^n) \le n \cdot (\mathrm{H}(X) - \varepsilon)\right] < 2 \cdot e^{-2\varepsilon^2 n}$.

Proof: (quantitative) AEP.

- $A_{n,\varepsilon} := \{\mathbf{x} \in \mathrm{Supp}(X^n) \colon 2^{-n(H(X)+\varepsilon)} \le p^n(\mathbf{x}) \le 2^{-n(H(X)-\varepsilon)}\}$
- $-\log p^n(\mathbf{x}) \ge n \cdot (\mathrm{H}(X) - \varepsilon)$ for any $\mathbf{x} \in A_{n,\varepsilon}$

Proposition 6 (Hoeffding's inequality)

Let $Z^1, \ldots, Z^n$ be iids over $[0,1]$ with expectation μ. Then, $\Pr\left[\left|\frac{\sum_{j=i}^n Z^j}{n} - \mu\right| \ge \varepsilon\right] \le 2 \cdot e^{-2\varepsilon^2 n}$ for every $\varepsilon > 0$.

- Taking $Z_i = \log p(X_i)$, it follows that $\Pr\left[X^n \notin A_{n,\varepsilon}\right] \le 2 \cdot e^{-2\varepsilon^2 n}$

Corollary 7

$\exists$ rv W that is $(2 \cdot e^{-2\varepsilon^2 n})$-close to X^n, and $\mathrm{H}_\infty(W) \ge n(H(X) - \varepsilon)$.

Proof: $W = X$ if $X \in A_{n,\varepsilon}$, and "well spread" outside $\mathrm{Supp}(X)$ otherwise.

Shannon to Min entropy, conditional version

Shannon to Min entropy, conditional version

Lemma 8

Let $(X, Y) \sim p$ *let* $\varepsilon > 0$. *Then*
$\Pr_{(x^n,y^n)\leftarrow(X,Y)^n}\left[-\log p^n_{X^n|Y^n}(x^n|y^n) \leq n\cdot(\mathrm{H}(X|Y)-\varepsilon)\right] < 2\cdot e^{-2\varepsilon^2 n}$.

Shannon to Min entropy, conditional version

Lemma 8

Let $(X, Y) \sim p$ *let* $\varepsilon > 0$. *Then*

$$\Pr_{(x^n,y^n)\leftarrow(X,Y)^n}\left[-\log p^n_{X^n|Y^n}(x^n|y^n) \leq n\cdot(\mathrm{H}(X|Y)-\varepsilon)\right] < 2\cdot e^{-2\varepsilon^2 n}.$$

Proof:

Shannon to Min entropy, conditional version

Lemma 8

Let $(X, Y) \sim p$ *let* $\varepsilon > 0$. *Then*
$\Pr_{(x^n,y^n)\leftarrow(X,Y)^n}\left[-\log p^n_{X^n|Y^n}(x^n|y^n) \leq n\cdot(\mathrm{H}(X|Y)-\varepsilon)\right] < 2\cdot e^{-2\varepsilon^2 n}$.

Proof: same proof, letting $Z_i = \log p_{X|Y}(X_i, Y_i)$

Shannon to Min entropy, conditional version

Lemma 8

Let $(X, Y) \sim p$ *let* $\varepsilon > 0$. *Then*
$\Pr_{(x^n,y^n)\leftarrow(X,Y)^n}\left[-\log p^n_{X^n|Y^n}(x^n|y^n) \leq n\cdot(\mathrm{H}(X|Y)-\varepsilon)\right] < 2\cdot e^{-2\varepsilon^2 n}$.

Proof: same proof, letting $Z_i = \log p_{X|Y}(X_i, Y_i)$

Corollary 9

$\exists$ *rv* W *over* $\mathcal{X}^n \times \mathcal{Y}^n$ *that is* $(2\cdot e^{-2\varepsilon^2 n})$*-far from* $(X, Y)^n$,

- $\mathrm{SD}(W_{\mathcal{Y}^n}, Y^n) = 0$, *and*

Shannon to Min entropy, conditional version

Lemma 8

Let $(X,Y) \sim p$ *let* $\varepsilon > 0$*. Then*
$\Pr_{(x^n,y^n)\leftarrow(X,Y)^n}\left[-\log p^n_{X^n|Y^n}(x^n|y^n) \leq n\cdot(\mathrm{H}(X|Y)-\varepsilon)\right] < 2\cdot e^{-2\varepsilon^2 n}$.

Proof: same proof, letting $Z_i = \log p_{X|Y}(X_i, Y_i)$

Corollary 9

$\exists$ *rv* W *over* $\mathcal{X}^n \times \mathcal{Y}^n$ *that is* $(2\cdot e^{-2\varepsilon^2 n})$*-far from* $(X,Y)^n$*,*

- $\mathrm{SD}(W_{\mathcal{Y}^n}, Y^n) = 0$*, and*
- $H(W \mid W_{\mathcal{Y}^n} = \boldsymbol{y}) \geq n\cdot(\mathrm{H}(X|Y)-\varepsilon)$*, for any* $\boldsymbol{y} \in \mathrm{Supp}(Y^n)$

Shannon to Min entropy, conditional version

Lemma 8

Let $(X,Y) \sim p$ *let* $\varepsilon > 0$. *Then*
$\Pr_{(x^n,y^n)\leftarrow(X,Y)^n}\left[-\log p^n_{X^n|Y^n}(x^n|y^n) \leq n\cdot(\mathrm{H}(X|Y)-\varepsilon)\right] < 2\cdot e^{-2\varepsilon^2 n}$.

Proof: same proof, letting $Z_i = \log p_{X|Y}(X_i,Y_i)$

Corollary 9

$\exists$ *rv* W *over* $\mathcal{X}^n \times \mathcal{Y}^n$ *that is* $(2\cdot e^{-2\varepsilon^2 n})$*-far from* $(X,Y)^n$,

- $\mathrm{SD}(W_{\mathcal{Y}^n}, Y^n) = 0$, *and*
- $H(W \mid W_{\mathcal{Y}^n} = \boldsymbol{y}) \geq n\cdot(\mathrm{H}(X|Y)-\varepsilon)$, *for any* $\boldsymbol{y} \in \mathrm{Supp}(Y^n)$

Shannon to Min entropy, conditional version

Lemma 8

Let $(X,Y) \sim p$ *let* $\varepsilon > 0$. *Then*
$\Pr_{(x^n,y^n)\leftarrow(X,Y)^n}\left[-\log p^n_{X^n|Y^n}(x^n|y^n) \leq n\cdot(\mathrm{H}(X|Y)-\varepsilon)\right] < 2\cdot e^{-2\varepsilon^2 n}$.

Proof: same proof, letting $Z_i = \log p_{X|Y}(X_i, Y_i)$

Corollary 9

$\exists$ *rv* W *over* $\mathcal{X}^n \times \mathcal{Y}^n$ *that is* $(2\cdot e^{-2\varepsilon^2 n})$*-far from* $(X,Y)^n$,

- $\mathrm{SD}(W_{\mathcal{Y}^n}, Y^n) = 0$, *and*
- $H(W \mid W_{\mathcal{Y}^n} = \boldsymbol{y}) \geq n\cdot(\mathrm{H}(X|Y)-\varepsilon)$, *for any* $\boldsymbol{y} \in \mathrm{Supp}(Y^n)$

Proof: ?

Section 2

Min-entropy to Uniform

Pairwise independent hashing

Definition 10 (pairwise independent function family)

A function family $\mathcal{G} = \{g \colon \mathcal{D} \mapsto \mathcal{R}\}$ is pairwise independent, if $\forall\ x \neq x' \in \mathcal{D}$ and $y, y' \in \mathcal{R}$, it holds that $\Pr_{g \leftarrow \mathcal{G}}\left[g(x) = y \wedge g(x') = y')\right] = (\frac{1}{|\mathcal{R}|})^2$.

Pairwise independent hashing

Definition 10 (pairwise independent function family)

A function family $\mathcal{G} = \{g \colon \mathcal{D} \mapsto \mathcal{R}\}$ is pairwise independent, if $\forall\, x \neq x' \in \mathcal{D}$ and $y, y' \in \mathcal{R}$, it holds that $\Pr_{g \leftarrow \mathcal{G}}\left[g(x) = y \wedge g(x') = y')\right] = (\frac{1}{|\mathcal{R}|})^2$.

- Example: for $\mathcal{D} = \{0,1\}^n$ and $\mathcal{R} = \{0,1\}^m$ let $\mathcal{G} = \{(A, b) \in \{0,1\}^{m \times n} \times \{0,1\}^m\}$ with $(A, b)(x) = A \times x + b$.

Pairwise independent hashing

Definition 10 (pairwise independent function family)

A function family $\mathcal{G} = \{g \colon \mathcal{D} \mapsto \mathcal{R}\}$ is pairwise independent, if $\forall\, x \neq x' \in \mathcal{D}$ and $y, y' \in \mathcal{R}$, it holds that $\Pr_{g \leftarrow \mathcal{G}}[g(x) = y \wedge g(x') = y')] = (\frac{1}{|\mathcal{R}|})^2$.

- Example: for $\mathcal{D} = \{0,1\}^n$ and $\mathcal{R} = \{0,1\}^m$ let $\mathcal{G} = \{(A, b) \in \{0,1\}^{m \times n} \times \{0,1\}^m\}$ with $(A, b)(x) = A \times x + b$.

Pairwise independent hashing

Definition 10 (pairwise independent function family)

A function family $\mathcal{G} = \{g \colon \mathcal{D} \mapsto \mathcal{R}\}$ is pairwise independent, if $\forall\ x \neq x' \in \mathcal{D}$ and $y, y' \in \mathcal{R}$, it holds that $\Pr_{g \leftarrow \mathcal{G}}\left[g(x) = y \wedge g(x') = y')\right] = (\frac{1}{|\mathcal{R}|})^2$.

- Example: for $\mathcal{D} = \{0,1\}^n$ and $\mathcal{R} = \{0,1\}^m$ let $\mathcal{G} = \{(A,b) \in \{0,1\}^{m \times n} \times \{0,1\}^m\}$ with $(A,b)(x) = A \times x + b$.
- 2-universal families: $\Pr_{g \leftarrow \mathcal{G}}\left[g(x) = g(x'))\right] = \frac{1}{|\mathcal{R}|}$.

Pairwise independent hashing

Definition 10 (pairwise independent function family)

A function family $\mathcal{G} = \{g \colon \mathcal{D} \mapsto \mathcal{R}\}$ is pairwise independent, if $\forall\ x \neq x' \in \mathcal{D}$ and $y, y' \in \mathcal{R}$, it holds that $\Pr_{g \leftarrow \mathcal{G}}\left[g(x) = y \wedge g(x') = y')\right] = (\frac{1}{|\mathcal{R}|})^2$.

- Example: for $\mathcal{D} = \{0,1\}^n$ and $\mathcal{R} = \{0,1\}^m$ let $\mathcal{G} = \{(A,b) \in \{0,1\}^{m \times n} \times \{0,1\}^m\}$ with $(A,b)(x) = A \times x + b$.
- 2-universal families: $\Pr_{g \leftarrow \mathcal{G}}\left[g(x) = g(x'))\right] = \frac{1}{|\mathcal{R}|}$.
- Example for universal family that is not pairwise independent?

Pairwise independent hashing

Definition 10 (pairwise independent function family)

A function family $\mathcal{G} = \{g \colon \mathcal{D} \mapsto \mathcal{R}\}$ is pairwise independent, if $\forall\ x \neq x' \in \mathcal{D}$ and $y, y' \in \mathcal{R}$, it holds that $\Pr_{g \leftarrow \mathcal{G}}\left[g(x) = y \wedge g(x') = y')\right] = (\frac{1}{|\mathcal{R}|})^2$.

- Example: for $\mathcal{D} = \{0,1\}^n$ and $\mathcal{R} = \{0,1\}^m$ let $\mathcal{G} = \{(A, b) \in \{0,1\}^{m \times n} \times \{0,1\}^m\}$ with $(A, b)(x) = A \times x + b$.
- 2-universal families: $\Pr_{g \leftarrow \mathcal{G}}\left[g(x) = g(x'))\right] = \frac{1}{|\mathcal{R}|}$.
- Example for universal family that is not pairwise independent?
- Many-wise independent

Pairwise independent hashing

Definition 10 (pairwise independent function family)

A function family $\mathcal{G} = \{g \colon \mathcal{D} \mapsto \mathcal{R}\}$ is pairwise independent, if $\forall\ x \neq x' \in \mathcal{D}$ and $y, y' \in \mathcal{R}$, it holds that $\Pr_{g \leftarrow \mathcal{G}}[g(x) = y \wedge g(x') = y')] = (\frac{1}{|\mathcal{R}|})^2$.

- Example: for $\mathcal{D} = \{0,1\}^n$ and $\mathcal{R} = \{0,1\}^m$ let $\mathcal{G} = \{(A, b) \in \{0,1\}^{m \times n} \times \{0,1\}^m\}$ with $(A, b)(x) = A \times x + b$.
- 2-universal families: $\Pr_{g \leftarrow \mathcal{G}}[g(x) = g(x'))] = \frac{1}{|\mathcal{R}|}$.
- Example for universal family that is not pairwise independent?
- Many-wise independent
- We identify functions with their description.

Pairwise independent hashing

Definition 10 (pairwise independent function family)

A function family $\mathcal{G} = \{g\colon \mathcal{D} \mapsto \mathcal{R}\}$ is pairwise independent, if $\forall\ x \neq x' \in \mathcal{D}$ and $y, y' \in \mathcal{R}$, it holds that $\Pr_{g \leftarrow \mathcal{G}}[g(x) = y \wedge g(x') = y')] = (\frac{1}{|\mathcal{R}|})^2$.

- Example: for $\mathcal{D} = \{0,1\}^n$ and $\mathcal{R} = \{0,1\}^m$ let $\mathcal{G} = \{(A, b) \in \{0,1\}^{m \times n} \times \{0,1\}^m\}$ with $(A, b)(x) = A \times x + b$.
- 2-universal families: $\Pr_{g \leftarrow \mathcal{G}}[g(x) = g(x'))] = \frac{1}{|\mathcal{R}|}$.
- Example for universal family that is not pairwise independent?
- Many-wise independent
- We identify functions with their description.
- Amazingly useful tool

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g \colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then
$\mathrm{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g \colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathrm{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g \colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathrm{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathrm{SD}(p, \sim \mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathsf{H}_2(X) \geq k$ let $\mathcal{G} = \{g \colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then
$\mathsf{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathsf{SD}(p, \sim \mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Proof: Let q be the uniform distribution over $\mathcal{U}$.

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g \colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathrm{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathrm{SD}(p, \sim \mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Proof: Let q be the uniform distribution over $\mathcal{U}$.

- $\|p - q\|_2^2 = \sum_{u \in \mathcal{U}} (d(u) - q(u))^2$

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g\colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathrm{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathrm{SD}(p, \sim \mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Proof: Let q be the uniform distribution over $\mathcal{U}$.

- $\|p - q\|_2^2 = \sum_{u \in \mathcal{U}} (d(u) - q(u))^2$

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g \colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathrm{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathrm{SD}(p, \sim \mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Proof: Let q be the uniform distribution over $\mathcal{U}$.

- $\|p - q\|_2^2 = \sum_{u \in \mathcal{U}} (d(u) - q(u))^2 = \|p\|_2^2 + \|q\|_2^2 - 2\langle p, q \rangle$

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathsf{H}_2(X) \geq k$ let $\mathcal{G} = \{g\colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathsf{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathsf{SD}(p, \sim \mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Proof: Let q be the uniform distribution over $\mathcal{U}$.

- $\|p - q\|_2^2 = \sum_{u \in \mathcal{U}} (d(u) - q(u))^2 = \|p\|_2^2 + \|q\|_2^2 - 2\langle p, q\rangle = \mathrm{CP}(p) - \frac{1}{|\mathcal{U}|} \leq \frac{\delta}{|\mathcal{U}|}$
- Chebyshev Sum Inequality: $(\sum_{i=1}^n a_i)^2 \leq n \sum_{i=1}^n a_i^2$

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g\colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathrm{SD}((G, G(X)), (G, \sim\{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathrm{SD}(p, \sim\mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Proof: Let q be the uniform distribution over $\mathcal{U}$.

- $\|p - q\|_2^2 = \sum_{u \in \mathcal{U}}(d(u) - q(u))^2 = \|p\|_2^2 + \|q\|_2^2 - 2\langle p, q\rangle = \mathrm{CP}(p) - \frac{1}{|\mathcal{U}|} \leq \frac{\delta}{|\mathcal{U}|}$
- Chebyshev Sum Inequality: $(\sum_{i=1}^n a_i)^2 \leq n \sum_{i=1}^n a_i^2$
- Hence, $\|p - q\|_1^2 \leq |\mathcal{U}| \cdot \|p - q\|_2^2$

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g \colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathrm{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathrm{SD}(p, \sim \mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Proof: Let q be the uniform distribution over $\mathcal{U}$.

- $\|p-q\|_2^2 = \sum_{u \in \mathcal{U}} (d(u)-q(u))^2 = \|p\|_2^2 + \|q\|_2^2 - 2\langle p, q\rangle = \mathrm{CP}(p) - \frac{1}{|\mathcal{U}|} \leq \frac{\delta}{|\mathcal{U}|}$
- Chebyshev Sum Inequality: $(\sum_{i=1}^n a_i)^2 \leq n \sum_{i=1}^n a_i^2$
- Hence, $\|p-q\|_1^2 \leq |\mathcal{U}| \cdot \|p-q\|_2^2$
- Hence, $\mathrm{SD}(p,q) = \frac{1}{2} \|p-q\|_1^2 \leq \frac{\sqrt{\delta}}{2}$.$\square$

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g\colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathrm{SD}((G, G(X)),(G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathrm{SD}(p, \sim \mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Proof: Let q be the uniform distribution over $\mathcal{U}$.

- $\|p-q\|_2^2 = \sum_{u\in\mathcal{U}}(d(u)-q(u))^2 = \|p\|_2^2 + \|q\|_2^2 - 2\langle p,q\rangle = \mathrm{CP}(p) - \frac{1}{|\mathcal{U}|} \leq \frac{\delta}{|\mathcal{U}|}$
- Chebyshev Sum Inequality: $(\sum_{i=1}^n a_i)^2 \leq n\sum_{i=1}^n a_i^2$
- Hence, $\|p-q\|_1^2 \leq |\mathcal{U}| \cdot \|p-q\|_2^2$
- Hence, $\mathrm{SD}(p,q) = \frac{1}{2}\|p-q\|_1^2 \leq \frac{\sqrt{\delta}}{2}$. $\square$

Leftover hash lemma

Lemma 11 (leftover hash lemma)

Let X be a rv over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$ let $\mathcal{G} = \{g\colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be 2-universal and let $G \leftarrow \mathcal{G}$. Then $\mathrm{SD}((G, G(X)), (G, \sim \{0,1\}^m)) \leq \frac{1}{2} \cdot 2^{(m-k))/2}$.

Extraction.

Lemma 12

Let p be a distribution over $\mathcal{U}$ with $CP(p) \leq \frac{1+\delta}{|\mathcal{U}|}$, then $\mathrm{SD}(p, \sim \mathcal{U}) \leq \frac{\sqrt{\delta}}{2}$.

Proof: Let q be the uniform distribution over $\mathcal{U}$.

- $\|p-q\|_2^2 = \sum_{u \in \mathcal{U}} (d(u) - q(u))^2 = \|p\|_2^2 + \|q\|_2^2 - 2\langle p, q\rangle = \mathrm{CP}(p) - \frac{1}{|\mathcal{U}|} \leq \frac{\delta}{|\mathcal{U}|}$
- Chebyshev Sum Inequality: $(\sum_{i=1}^n a_i)^2 \leq n \sum_{i=1}^n a_i^2$
- Hence, $\|p-q\|_1^2 \leq |\mathcal{U}| \cdot \|p-q\|_2^2$
- Hence, $\mathrm{SD}(p,q) = \frac{1}{2}\|p-q\|_1^2 \leq \frac{\sqrt{\delta}}{2}$.$\square$

To deuce the proof of Lemma 11, we notice that $\mathrm{CP}(G, G(X)) \leq \frac{1}{|\mathcal{G}|} \cdot (2^{-k} + 2^{-m}) = \frac{1+2^{m-k}}{|\mathcal{G} \times \{0,1\}^n|}$